Package: SARIMAInfoRF (via r-universe)

November 20, 2026

Type Package

Title SARIMA-Informed Random Forest for Time Series Forecasting

Version 0.1.0

Description Trains a Random Forest on a target blended with the forecasts of a seasonal autoregressive integrated moving average (SARIMA) model, which acts as a teacher. For squared error, the composite loss (1 - lambda) * MSE(f, y) + lambda * MSE(f, y_SARIMA), where MSE is the mean squared error, is minimised exactly by fitting the forest to the blended target z = (1 - lambda) * y + lambda * y_SARIMA. The SARIMA model is chosen automatically by the Akaike information criterion (AIC) or specified manually; the forest hyperparameters are set manually, by grid search, or by Bayesian optimisation. The best lambda is chosen on a chronological validation period. Functions for out-of-sample forecasting, accuracy measures and plots are provided. The methodology is motivated by hybrid forecasting framework proposed by Yeasin and Paul (2024) <doi:10.1007/s11227-023-05542-3>.

License MIT + file LICENSE

Encoding UTF-8

Imports ranger, forecast, ggplot2, cli, stats, utils

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Ranjit Kumar Paul [aut, cre], Pushkar Bora [aut], Md Yeasin [aut]

Maintainer Ranjit Kumar Paul <ranjitstat@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-20 13:12:36 UTC

RemoteUrl https://github.com/cran/SARIMAInfoRF

RemoteRef HEAD

RemoteSha dbe1ab95b951d680d430a7ce33c20b4b4a123f65

Contents

AccuracySiRF 2
plotlambda_sirf 3
plotSiRF 3
predictSiRF 5
SiRF 6

Index **10**

`AccuracySiRF` *Out-of-sample accuracy of SiRF forecasts against actual values*

Description

Forecasts `length(actual)` steps ahead from the end of the training data and compares the forecasts with the values that actually arrived. The SARIMA model is evaluated alongside as a reference.

Usage

```
AccuracySiRF(fit, actual, lambda = NULL, verbose = TRUE)
```

Arguments

`fit`	A fitted `SiRF` object.
`actual`	Numeric vector of observations AFTER the training data.
`lambda`	Lambda to use; `NULL` = best lambda.
`verbose`	Print the accuracy report to the console. Default `TRUE`; use `FALSE` to suppress it.

Value

Invisibly, a data frame (Model, Lambda, h, RMSE, MAE, SMAPE, MAPE) with one row for SiRF and one for SARIMA. A step-by-step table (Actual, SiRF, SARIMA, Error, Abs_Error) is in attribute `"comparison"`.

Examples

```
set.seed(1)
y <- 100 * exp(cumsum(rnorm(108, 0.004, 0.04)) +
               0.15 * sin(2 * pi * (1:108) / 12))
fit <- SiRF(y[1:96], lag = 6, sarima_mode = "manual", teacher = "fitted",
            rf_params = list(n_estimators = 50), lambda_list = 0.5,
            verbose = FALSE, seed = 1)
acc <- AccuracySiRF(fit, actual = y[97:108])
attr(acc, "comparison")
```

plotlambda_sirf *Plot the selection RMSE by lambda (bar chart)*

Description

Bar chart of the RMSE used to choose lambda (validation RMSE by default) for the plain forest (lambda = 0) and every lambda. The best lambda is the gold bar.

Usage

```
plotlambda_sirf(x, title = NULL)
```

Arguments

x	A fitted SiRF object.
title	Optional title.

Value

A ggplot2 object (printed, and returned invisibly).

Examples

```
set.seed(1)
y <- 100 * exp(cumsum(rnorm(96, 0.004, 0.04)) +
               0.15 * sin(2 * pi * (1:96) / 12))
fit <- SiRF(y, lag = 6, sarima_mode = "manual", teacher = "fitted",
            rf_params = list(n_estimators = 50), lambda_list = c(0.3, 0.7),
            verbose = FALSE, seed = 1)
plotlambda_sirf(fit)
```

plotSiRF *Plot actual, fitted and forecast values for a fitted SiRF model*

Description

Full actual series (dark), in-sample fitted values (blue) and the h-step forecast (red); optionally the SARIMA forecast (dashed) and the actual future values (green).

Usage

```
plotSiRF(
  x,
  h = 12,
  lambda = NULL,
  actual = NULL,
  newdata = NULL,
  title = NULL,
  n_hist = NULL,
  date = NULL,
  show_sarima = TRUE,
  ...
)
```

Arguments

x	A fitted SiRF object.
h	Forecast horizon. Default 12 (extended to length(actual) when actual is longer).
lambda	Lambda to use; NULL = best lambda.
actual	Optional observed values after the training data (green line).
newdata	Optional new observations appended before forecasting.
title	Optional title.
n_hist	Show only the last n_hist in-sample points.
date	Optional Date/POSIXct vector for the training data. Defaults to the date stored in the object; otherwise a numeric index is used.
show_sarima	Draw the SARIMA forecast as a dashed line. Default TRUE.
...	Ignored.

Details

plot(x, ...) on a fitted SiRF object is equivalent to plotSiRF(x, ...).

Value

A ggplot2 object (printed, and returned invisibly).

Examples

```
set.seed(1)
y <- 100 * exp(cumsum(rnorm(96, 0.004, 0.04)) +
               0.15 * sin(2 * pi * (1:96) / 12))
fit <- SiRF(y, lag = 6, sarima_mode = "manual", teacher = "fitted",
            rf_params = list(n_estimators = 50), lambda_list = 0.5,
            verbose = FALSE, seed = 1)
plotSiRF(fit, h = 12)
```

predictSiRF *Forecast future values using a fitted SiRF model*

Description

Recursive h-step point forecasts from the forest (the SARIMA teacher is not used). The SARIMA model's own forecasts are returned as a reference. No confidence intervals are produced.

Usage

```
predictSiRF(object, h = 1, lambda = NULL, newdata = NULL, ...)
```

Arguments

object	A fitted SiRF object.
h	Number of steps ahead. Default 1.
lambda	Lambda to use; NULL = best lambda.
newdata	Optional numeric vector of observations that arrived after training; the forecast starts after them.
...	Ignored.

Details

predict(object, h, ...) is equivalent to predictSiRF(object, h, ...). Printing the returned SiRF_forecast object shows a table of the forecasts.

Value

An object of class SiRF_forecast: a list with forecast, sarima_forecast, time_idx, h, lambda and sarima_model.

Examples

```
set.seed(1)
y <- 100 * exp(cumsum(rnorm(96, 0.004, 0.04)) +
               0.15 * sin(2 * pi * (1:96) / 12))
fit <- SiRF(y, lag = 6, sarima_mode = "manual", teacher = "fitted",
            rf_params = list(n_estimators = 50), lambda_list = 0.5,
            verbose = FALSE, seed = 1)
predictSiRF(fit, h = 6)
```

SiRF *SARIMA-Informed Random Forest (SiRF)*

Description

Trains a Random Forest on a target blended with the forecasts of a SARIMA teacher. For squared error the composite loss

```
L = (1 - lambda) * MSE(f, y) + lambda * MSE(f, y_SARIMA)
```

equals `(f - z)^2` plus a term that does not depend on the forecast `f`, where `z = (1 - lambda) * y + lambda * y_SARIMA`. Fitting the forest (whose trees minimise squared error) to `z` therefore minimises exactly this loss. `lambda = 0` is a plain Random Forest.

Usage

```
SiRF(
  data,
  date = NULL,
  lag,
  frequency = 12L,
  start_season = 1L,
  log_transform = TRUE,
  give_season = TRUE,
  sarima_mode = c("auto", "manual"),
  sarima_order = c(1, 1, 1),
  sarima_seasonal = c(0, 1, 1),
  sarima_max = list(p = 2, d = 1, q = 2, P = 1, D = 1, Q = 1),
  teacher = c("rolling", "fitted"),
  teacher_refit_every = 1L,
  min_seasonal_obs = 36L,
  rf_mode = c("manual", "grid", "bayes"),
  rf_params = list(),
  rf_grid = list(),
  rf_space = list(),
  n_init = 10L,
  n_iter = 30L,
  max_combos = 100L,
  lambda_list = seq(0.1, 0.9, 0.1),
  lambda_select = c("validation", "train"),
  val_frac = 0.2,
  verbose = TRUE,
  seed = NULL
)
```

Arguments

data	Numeric vector (the time series).
date	Optional Date/POSIXct vector (same length as data); used to find the calendar month for the season inputs and for plot axes.
lag	Number of past values used as input. **Required, no default.** A single value fixes it. With rf_mode = "grid" a vector is the set of candidates; with "bayes" a vector c(min, max) is the search range.
frequency	Seasonal period (12 monthly, 4 quarterly, 1 none). Default 12.
start_season	Season position (1..frequency) of the first observation, used only when date does not give the month. Default 1.
log_transform	Model the log of the data (as in the reference implementation). Switched off with a warning if the data contain values <= 0. Default TRUE.
give_season	Give the season position (sine, cosine) to the forest. Default TRUE.
sarima_mode	"auto" (AIC) or "manual".
sarima_order	Manual mode: c(p, d, q).
sarima_seasonal	Manual mode: c(P, D, Q).
sarima_max	Auto mode: named list of the largest orders searched (p, d, q, P, D, Q); defaults 2, 1, 2, 1, 1, 1.
teacher	"rolling" (default) or "fitted".
teacher_refit_every	Rolling teacher: re-estimate parameters every this many months (others reuse the last parameters). Default 1.
min_seasonal_obs	Rolling teacher: below this many observations of history the seasonal part is dropped. Default 36.
rf_mode	"manual", "grid" or "bayes".
rf_params	Manual mode: list with n_estimators, max_depth, min_node_size, max_features (share of inputs tried per split). Missing entries take defaults (300, 10, 3, 0.7).
rf_grid	Grid mode: named list of candidate vectors for the same four settings (missing ones use defaults).
rf_space	Bayes mode: named list of c(min, max) ranges for the same four settings (missing ones use defaults).
n_init, n_iter	Bayes mode: initial random evaluations and further iterations (per lambda). Defaults 10 and 30.
max_combos	Grid mode: if the grid is larger, this many randomly chosen combinations are evaluated. Default 100.
lambda_list	Lambda values to run. Lambda 0 (plain Random Forest) is always added. Default seq(0.1, 0.9, 0.1).
lambda_select	How the best lambda is chosen: "validation" (default; recommended) or "train" (in-sample RMSE, which always favours small lambda for a flexible forest).

val_frac	Share of the series used as validation period. Default 0.2.
verbose	Print progress messages. Default TRUE.
seed	Optional integer random seed for reproducible results. Default NULL: no seed is set (use set.seed() beforehand to make a run reproducible).

Details

The model works on the (log) series. Inputs are the last lag values relative to the latest value, divided by the typical one-step change, plus the season position of the target month (sine and cosine). The target is the next change. The SARIMA teacher is used for training only; forecasting uses the forest alone.

SARIMA is chosen automatically by AIC (sarima_mode = "auto") or specified by the user ("manual"). **Teacher forecasts** are either "rolling" (parameters re-estimated every month with the history up to that month, so the teacher's forecasts for the training months are genuine out-of-sample forecasts) or "fitted" (one-step fitted values of the full-data model; much faster).

Random Forest hyperparameters are set manually (rf_mode = "manual"), by grid search ("grid") or by Bayesian optimisation ("bayes"; Gaussian process with expected improvement). Tuning is done separately for every lambda on a chronological validation period (the last val_frac of the series); the final forest is then refitted on all data. The best lambda is chosen on the validation period.

The returned object has methods for the usual generics: print(fit) gives a short report; summary(fit) gives a detailed report (SARIMA model, forest hyperparameters and the results for every lambda); plot(fit) and predict(fit) are equivalent to plotSiRF and predictSiRF.

Value

An object of class "SiRF" with components predictions (actual and fitted values of the best lambda), metrics (in-sample RMSE, MAE, MAPE, SMAPE, plus validation RMSE), lambda_results, best_lambda, best_hp, sarima_info, teacher_forecasts, scaler, train_time, meta and data_info.

Examples

```
set.seed(1)
y <- 100 * exp(cumsum(rnorm(96, 0.004, 0.04)) +
               0.15 * sin(2 * pi * (1:96) / 12))
fit <- SiRF(y, lag = 6, sarima_mode = "manual", teacher = "fitted",
            rf_mode = "manual", rf_params = list(n_estimators = 50),
            lambda_list = 0.5, verbose = FALSE, seed = 1)
fit
summary(fit)


# Automatic SARIMA selection and grid search for the forest (slower)
fit2 <- SiRF(y, lag = c(3, 6), sarima_mode = "auto",
             sarima_max = list(p = 1, d = 1, q = 1, P = 1, D = 1, Q = 1),
             teacher = "fitted", rf_mode = "grid",
             rf_grid = list(n_estimators = 50, max_depth = c(5, 10),
                            min_node_size = 3, max_features = 0.7),
             lambda_list = c(0.3, 0.7), verbose = FALSE, seed = 1)
```

```
fit2$best_hp
plotlambda_sirf(fit2)
```

Index

AccuracySiRF, 2

plotlambda_sirf, 3
plotSiRF, 3, *8*
predictSiRF, 5, *8*

SiRF, 6